Bayesian Methods for Assessing System Reliability: Models and Computation

Todd L. Graves
Statistical Sciences
Los Alamos National Laboratory
Los Alamos, New Mexico
USA
tgraves@lanl.gov

Michael S. Hamada
Statistical Sciences
Los Alamos National Laboratory
Los Alamos, New Mexico
USA
hamada@lanl.gov

Abstract

There are many challenges with assessing the reliability of a system today. These challenges arise because a system may be aging and full system tests may be too expensive or can no longer be performed. Without full system testing, one must integrate (1) all science and engineering knowledge, models and simulations, (2) information and data at various levels of the system, e.g., subsystems and components and (3) information and data from similar systems, subsystems and components. The analyst must work with various data types and how the data are collected, account for measurement bias and uncertainty, deal with model and simulation uncertainty and incorporate expert knowledge.

Bayesian hierarchical modeling provides a rigorous way to combine information from multiple sources and different types of information. However, an obstacle to applying Bayesian methods is the need to develop new software to analyze novel statistical models. We discuss a new statistical modeling environment, YADAS, that facilitates the development of Bayesian statistical analyses. It includes classes that help analysts specify new models, as well as classes that support the creation of new analysis algorithms. We illustrate these concepts using several real-world examples.

1 Challenges in Modern Reliability Analyses

There are many challenges with assessing the reliability of a system today. First, full system testing may be prohibitively expensive or even prohibited. For this reason and others, it is important to be able to make use of expert opinion and information in the form of physics/engineering/material science based models (deterministic and stochastic) or simulation and account for model bias and uncertainty. One must be able to handle complex system reliability models, including reliability block diagrams and fault trees. One must incorporate data at various levels (system, subsystem, component), and properly account for how higher level event data inform about lower level data. In this context, models for subsystems or even components can be non-trivial. The effects of aging and other covariates including those that define subpopulations, are often of interest. Efficient analysis entails combining information and data from similar systems, subsystems and components. Reliability data can come in various flavors, including binomial counts, Poisson counts, failure times, degradation data, and accelerated reliability data. Measurement error (bias and precision) from destructive or nondestructive evaluation techniques may be too large to ignore. In what follows we consider three examples which illustrate these challenges.

2 Three Important Examples

We discuss three examples of challenging statistical problems that arise in reliability estimation. First, even the analysis of a single component can require development of new techniques. Consider the case in which there are indications that a component's manufacturing lot impacts its reliability, and some of the test data are obtained in ways that might favor the sampling of (un)reliable items. Second, we discuss the estimation of the reliability of a system based on (1) system tests, where failures provide partial information about which components may have failed, and (2) specification tests, which measure whether components

meet specifications that relate imperfectly to system success. Finally, we present an ambitious approach to integrating many sorts of component data into a system reliability analysis.

2.1 Reliability of a Component Based on Biased Sampling

Our first example, which deals with reliability estimation for a single component, is discussed in Graves *et al* (2004). Of interest is the prevalence of a certain feature in an existing population of items. Some items have already been destructively tested and removed from the population. There is reason to believe that the probability that an item has the feature is related to the lot in which it was manufactured, but it is not obviously appropriate to assume that the feature is confined to a small number of lots. We handle this situation with a Bayesian hierarchical model, $p_i \sim \text{Beta}(a, b)$, where p_i is the probability that an item in lot i was manufactured with the feature, and where a and b are given prior distributions, so that a test on an item in one lot is informative about the prevalence of the feature in the other lots, but more informative about its own lot. A further complication is we are not willing to assume that the process by which items were selected for sampling was done so that items with and without the feature were equally likely to be sampled. Naive estimation is therefore in danger of systematically over- or under-estimating the prevalence. We use the *extended hypergeometric* distribution (see Graves *et al* (2004) and its references) to allow for the possibility of biased sampling; using this distribution for this purpose was new, so software did not exist for using it. Finally, the unknown quantities of most interest are the actual numbers of items with the feature in each of the lots (which were of known finite size), so the software must be able to sample posterior distributions of quantities which take on finitely many values.

2.2 System Reliability Based on Partially Informative Tests

Another reliability problem involves synthesizing two different types of data, neither of which is standard for reliability analysis. First, the system test data provide complicated information: for notational clarity, consider a single system test. If the set of components in the system is denoted by C, there is a subset of components C_1 that we know to have worked, another subset of components C_0 that we know to have failed, and a third subset of components C_2, at least one of which must have failed. (The test provides no information at all about the success or failure of the remaining components.) The system is a series system. The likelihood function for this single test, assuming that the system is of age t, is

$$\prod_{i \in C_1} p_i(t) \prod_{i \in C_0} \{1 - p_i(t)\} \left\{ 1 - \prod_{i \in C_2} p_i(t) \right\}, \tag{1}$$

where the first two products are defined to be one if empty, while the last is zero. Here $p_i(t)$ is the probability of success of component i at age t: we used $p_i(t) = \Phi\left((\sigma_i^2 + \gamma_i^2)^{-1/2}(\alpha_i + \beta_i t - \theta_i)\right)$, where Φ is the Gaussian distribution function.

One reason for this choice of $p_i(t)$ (in particular, the seemingly redundant parameterization) is the other type of data we use: certain of the components were tested to assure that they met specifications, and these tests generate continuous data. If one assumes that the a specification measurement S_i on component i satisfies $S_i \sim N(\alpha_i + \beta_i t, \gamma_i^2)$, specification data can be incorporated naturally. Then if one assumes that, conditionally on its specification measurement S_i, the component would succeed in a system test with probability $\Phi\left(\sigma_i^{-1}(S_i - \theta_i)\right)$, it follows that unconditionally, the component's success probability is $\Phi\left((\sigma_i^2 + \gamma_i^2)^{-1/2}(\alpha_i + \beta_i t - \theta_i)\right)$. This enables us to use specification data to help make inferences on parameters relevant to system tests. They also require special-purpose software.

2.3 Integrated System Reliability Based on Diverse Data

A fundamental problem of system reliability is estimating the reliability of a system whose components are combined in series and parallel subsystems, and where data relevant to the component qualities take on general (not necessarily binomial) forms. (The case of binomial data is discussed in Johnson *et al* (2003).) As a simple example, consider a three component series system. Binary data is available on Component 1 at various ages, and its success probability at age t satisfies $\log(p_1(t)/\{1 - p_1(t)\}) = \alpha_1 - \beta_1 t$. "Success"

for Component 2 is defined in terms of its lifetime which is distributed Weibull: a lifetime of greater than τ_2 equates to component success at time $t < \tau_2$, and data on Component 2 are a collection of possibly right-censored lifetimes. Component 3 is required to generate a desired amount of power on demand; the distribution of power is lognormal, with a logged mean that decreases linearly in age. Data are (power, age) pairs. Finally, we have binary system test data, where the success probability is the age-dependent success probability of Component 1, multiplied by the reliability of Component 2, multiplied by the age-dependent probability of sufficient power generation by Component 3. The full data likelihood contains terms for each of the four types of tests, and other information can be captured in prior distributions. It is necessary that the software analyze likelihood functions with each of these terms, and ideally it would support the integration of components into (sub)systems in arbitrary parallel/series combinations.

3 Useful Features of YADAS

YADAS is a software environment for performing arbitrary Markov chain Monte Carlo computations, and as such it is very useful for defining and analyzing new, nonstandard statistical models. Its source code and documentation, several examples, and supporting technical reports are available for download at `yadas.lanl.gov` (Graves (2001)). Its software architecture makes it easy to define new terms in models and make small adjustments to existing models. MCMC algorithms often suffer from poor autocorrelation, and YADAS provides an environment for exploring and fixing these problems. YADAS is written in Java and generally requires additional Java code to work a new problem, but work continues on alternative interfaces. We discuss all these issues in this section.

3.1 Expressing arbitrary models

Defining a model in YADAS is as simple as specifying how to calculate the unnormalized posterior distribution evaluated at an arbitrary parameter value. This is an advantage of a Bayesian approach, as well as being one of the benefits of the design decision to emphasize the Metropolis–Hastings algorithm instead of Gibbs sampling as in WinBUGS (Spiegelhalter *et al*, 2000). In the Gibbs sampler, each time the model is changed, the sampling algorithm must be changed accordingly. In YADAS, however, the model definition is decoupled from the algorithm definition. The correct posterior distribution will be sampled if the acceptance probabilities are generated correctly, and since the acceptance probabilities are determined by the unnormalized posterior density function, this happens automatically when the new model is defined.

The definition of a model is a collection of objects called *bonds*. Each bond is a term in the posterior distribution. Bonds are defined in the software in such a way as to make it easy to make the sort of small changes to an analysis that are common in the model–building phase. Examples include changing a parameter from fixed to random, or changing a distributional form.

The ease of defining new models was particularly evident in the analysis of the reliability of the component manufactured in lots and sampled nonrandomly. The analysts needed to write code to calculate the extended hypergeometric density function, but after this trivial exercise was complete, it could be plugged in as any other density would be. Without YADAS, time constraints would have forced the analysts to make use of a more convenient but less appropriate analysis.

In our second example, the critical step was to compute (1) after first computing the $p_i(t)$'s. This was also straightforward, and the handling of specification data just required adding another bond (with the familiar normal linear model form) to the existing list. The third example, in which various forms of component test data are combined with system test data, is an excellent example of the usefulness of the YADAS model definition strategy. The component test data are as easy to include as in an application where they are the only data source. We make use of YADAS's general code that reads in a system structure and integrates component success probabilities in any series or parallel combination, in order to incorporate the system test data.

3.2 Special algorithms

While it is true that defining an MCMC algorithm for a new problem is as easy as specifying how to compute the unnormalized posterior distribution, it is also true that these first attempts at algorithms may fail to

perform adequately. However, YADAS turns this to a strength by helping users to improve algorithms by adding steps to the existing algorithm; Metropolis-Hastings based software is much better suited to this goal than Gibbs-based software. The most common MCMC performance problem is high posterior correlation among parameters; this generates high autocorrelation in consecutive MCMC samples, because parameters are reluctant to move individually. YADAS's typical approach is the "multiple parameter update": one proposes simultaneous moves to parameters in a direction of high variability. For example, in our second example (as happens in many generalized linear model examples), the intercept and slope parameters for some components were highly correlated, and the algorithm was improved with new steps that proposed the addition of a random amount to the intercept while simultaneously subtracting a multiple of the same amount from the slope.

The naturalness of defining algorithms in YADAS was also exhibited in the biased sampling problem, where the numbers of items in each lot with the feature of interest needed to be sampled; this was handled with YADAS's general approach for sampling parameters that take on finitely many values.

3.3 Interfaces, present and future

YADAS is written in Java, and that provides portability advantages beyond its encouragement of generality that helped YADAS become as ambitious as it is. However, few Bayesian statisticians use Java as their language of choice, so this limits its popularity. An area of active YADAS development is providing additional interfaces to its capabilities. One such interface is the interface with the R package (`www.R-project.org`), a very popular, free statistics computing environment that is very similar to S-Plus. This interface facilitates the handling of both input to and output from MCMC algorithms. One application we are exploring is using genetic algorithms for experimental design: each candidate design selected by the genetic algorithm will generate data, which will then be analyzed using YADAS, and the analysis results will be examined for "fitness" and fedback to determine the next genetic algorithm generation. System reliability is an application of particular interest. The R-YADAS interface is possible thanks to the SJava package of the omegahat project (`www.omegahat.org`).

Another interface that will particularly help with reliability problems is the interface with a new graphical tool for eliciting defining system structure and its relationship to data (Klamann and Koehler, 2004).

Acknowledgments

We thank Dee Won for her encouragement of this work.

References

Graves, T.L. (2001). YADAS: an object-oriented framework for data analysis using Markov chain Monte Carlo. Los Alamos National Laboratory Technical Report LA-UR-01-4804.

Graves, T., Hamada, M., Booker, J., Decroix, M., Bowyer, C., Chilcoat, K., Thompson, S.K. (2004). Estimating a proportion using stratified data arising from both convenience and random samples. Los Alamos National Lab Technical Report LA-UR-03-8396.

Johnson, V., Graves, T., Hamada, M., Reese, C.S. (2003). A hierarchical model for estimating the reliability of complex systems (with discussion). *Bayesian Statistics 7*, Oxford University Press, 199-213, Bernardo, J.M., Bayarri, M.J., Berger, J., Dawid, A.P., Heckerman, D., Smith, A.F.M. and West, M. (Eds.).

Klamann, R. and Koehler, A. (2004). GROMIT: Graphical Modeling Tool for System Statistical Structure. Los Alamos, NM: Los Alamos National Laboratory.

Spiegelhalter, D., Thomas, A., Best, N. (2000). *WinBUGS Version 1.3 User Manual*.